FSF: Friendship and Selfishness Forwarding for Delay Tolerant Networks

Camilo Souza, Edjair Mota, Leandro Galvao
Federal University of Amazonas, Institute of Computing
Manaus, Brazil
Email: {camilo,edjair,galvao}@icomp.ufam.edu.br

Pietro Manzoni, Juan Carlos Cano, Carlos T. Calafate
Universitat Politècnica de València, Computer Engineering Department
Valencia, Spain
Email: {pmanzoni,jucano,calafate}@disca.upv.es

***Abstract*—This paper presents the friendship and selfishness forwarding (FSF) algorithm for Delay Tolerant Networks. This novel solution is based on two social characteristics of nodes: friendship and selfishness. When a contact opportunity arises, FSF analyzes two aspects to make message forwarding decisions: first, FSF assesses the friendship strength among the pair of nodes, then it determines the individual selfishness of the relay node.**

Unlike other algorithms proposed in the DTN literature, we use a machine learning algorithm to quantify the friendship strength among pairs of nodes in the network. The individual selfishness of the relay node is determined by using a model based on the current level of its resources. The primary goal is to take into account the case where, despite a strong friendship with the message destination, the relay node does not accept processing the message to save its resources. By using trace-driven simulations we show that the FSF algorithm achieves better results in terms of delivery rate, average cost and efficiency.

I. Introduction

Delay Tolerant Networking (DTN) has been drawing the attention of the scientific community during the last years. This networking paradigm addresses the problem of intermittent connections caused by link failures, network partitioning, topology changes, or node mobility. Routing is a problem common to all DTN solutions since efficient protocols are required to overcome the long delays and frequent disconnection problems [1]. Based on DTN, other connectivity approaches are appearing, like for example with the surge of "Opportunistic Networks" or "Location-Based Social Networks".

A simple way to carry out message delivery in a scenario is to combine the store-and-forward mechanism with the deployment of additional nodes (ferries) that can act as data mules [2]. In opportunistic networks, all nodes play the role of ferries, and the exchange of messages occurs during a contact, i.e., whenever two nodes are within communications range. A refinement of this approach is to select the next hop node based on the probability of message delivery to the destination node, or based on the encounters history of the receiving node [3] [4]. Location-Based Social Networks take into account the behaviour of users carrying mobile devices [2].

The design of DTN protocols can therefore conveniently consider some social characteristics like node popularity, centrality, selfishness, and altruism to decide whether or not to forward messages to a specific node. In this work, we are interested in DTN message routing by taking into account the friendship level among the nodes, as well as the individual selfishness of the message relay candidate. In the DTN literature, we can already find some works taking into account the friendship and/or selfishness among nodes [5], [6], [7]. In these works, different metrics are introduced based on characteristics related to the friendship and/or the selfishness of users in the real world.

In this paper we present the "Friendship and Selfishness Forwarding (FSF)", a routing algorithm for DTNs that provides a novel approach based on machine learning for classifying the friendship strength among two nodes. We consider that real devices have limited resources (battery, memory, ...), and it is therefore reasonable that, to save resources, they are willing to collaborate with just a limited number of people. We can assume that the messages will be forwarded during contact opportunities only when there is some relationship among the nodes. The stronger the social relationship, the higher will be the willingness to help each other [5]. Thus, to forward a message, FSF uses the social relationship strength evidenced by the friendship strength among two nodes. If the friendship strength among two nodes is high, the message can be forwarded to the message relay. However, FSF still needs to check the message relay for selfishness. Unlike other approaches in the DTN literature, we use a machine learning algorithm as a friendship strength oracle. More details on this issue are provided in section III.

To validate our proposal, we made a set of experiments that include simulating the message delivery process in a DTN scenario, and comparing the results in terms of delivery rate, average delay, cost and efficiency against other protocols found in the literature such as BUBBLE-RAP [8] and Friendship Routing [7].

The main contributions of this paper are the following:

- Evaluation of the user willingness to participate. Since

978-1-5090-0679-3/16/$31.00 ©2016 IEEE

two friends may not want or be able to carry each other's message, both the friendship strength and the user selfishness should be considered in the forwarding decision, and FSF is an attempt to close this gap in the literature.
- Dynamic classification of the friendship strength. FSF tries to adapt itself to the real conditions of each node during a contact. The Naive Bayes machine learning algorithm is the core of this search towards a more realistic routing protocol.

The rest of this paper is organised as follows: in Section II we discuss some related works. Section III details the proposed FSF protocol for DTN applications. In Section IV, we describe the evaluation methodology. In Section V, we present and discuss the experimental results. Finally, in Section VI, we conclude the paper and refer to future works.

II. Related Works

In the DTN literature we can find a large variety of routing protocol proposals. The survey conducted by Zhu et al. [2] classified those protocols into three categories: message-ferry-based, opportunity-based, and prediction-based. Recently, protocols based on social relationships emerged as a fourth category. The reasoning behind this research branch is that most DTN applications are formed by a huge number of mobile devices carried by humans, whose behavior is better described by social network models [2]. This category is a worthwhile contribution to leverage the DTN research field since social characteristics are less volatile than node mobility. Among the social characteristics worth considering we have centrality, popularity, similarity, friendship, and selfishness.

Similarly to our approach, [7] proposed a routing algorithm based on friendship. To model the friendship among nodes, the authors considered the frequency of contacts, their longevity, and their regularity. Two nodes are considered friends if they regularly have long and frequent contacts. The authors proposed a new metric called Social Pressure Metric (SPM) to represent the social pressure motivating friends to visit each other and share their experiences. Qin et al. [6] proposed improvements in the friendship-based routing by means of an analysis of the inherent drawbacks, and proposed a new algorithm that also considers the contact periods.

Both protocols represent the friendship based on average contact frequency, longevity, and regularity. In this work, we also model friendship by taking into account the average contact frequency and duration, but in addition we consider the number of calls, and text messages exchanged, since we believe that such information is a clear evidence of friendship among two nodes. We believe that, the more calls and/or messages are exchanged by two nodes, the stronger will be the friendship among them. Furthermore, our algorithm uses a machine learning method to classify friendship strength among two nodes based on real-world examples.

The participation of nodes in the routing process is taken for granted by most existing routing protocols [9]. However, in the real world, individuals are socially selfish and tend to collaborate only with a restrict set of people. The Social Selfishness Aware Routing (SSAR) [5] is a protocol that introduced selfishness considerations in DTN scenarios. By recognising that selfishness in unavoidable, SSAR pursues to compensate for performance loss by allocating resources (buffers and bandwidth) based on packet priority. The authors of that paper consider that selfishness issues should be integrated in new routing algorithms since people are socially selfish, that is, they are willing to forward messages to a limited number of persons, and this willingness depends on the strength of the relationship among them. For example, if the social bonding among node A and node B is strong, node A will always accept messages to node B. On the contrary, FSF considers that, in several situations, node A may not accept messages destined to B for the sake of saving resources for itself, no matter how strong is their mutual friendship. Another important issue is to evaluate the performance of routing algorithms aiming to tackle the impact of selfishness on routing. For more details on this subject please refer to [10], [11], [12].

III. The FSF algorithm

The proposed FSF algorithm makes message forwarding decisions based on two steps: (i) classification of the friendship strength among the nodes, and (ii) quantifications of the message relayer's individual selfishness. In this section, more details will be provided for these two steps.

A. Classification of the friendship strength

As detailed in the previous Section, most of the current works that take friendship into account for improving routing, typically rely on metrics based on some characteristics related to the friendship among people in the real world. For example, Bulut and Szymanski [7] model the friendship among nodes by considering the contact frequency, longevity, and regularity; these characteristics are the base for a metric called Social Pressure Metric (SPM).

In our work we rely on an oracle that, using a machine learning algorithm based on Naive Bayes [13], is able to classify the friendship level among users. To be effective this oracle first has to learn what is "friendship" in the real world. This was achieved by taking into account information from a real experiment, based on which it was possible to classify the friendship level among two nodes.

We choose Naive Bayes based on the performance reported in other works found in the literature and the nature of the problem itself, for which it is reasonable to assume that attributes are statistically independent [14], [15], [16]. For example, if two nodes called each other several times, it does not imply that they also exchanged several text messages. According to [16] and [17], the Naive Bayes Classifier achieves excellent results in situations of statistical independence of the characteristics used for the task of classification; in other words, it can correctly predict the class of a given instance.

1) Naive Bayes algorithm: According to [18], machine learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time. The Naive Bayes machine learning algorithm is a supervised classifier, meaning that it uses previous instances along with their attributes to classify new instances. In this context, an *instance* is a tuple containing the values for the considered attributes of the problem. The Naive Bayes classifier uses the Bayes' theorem to calculate the probabilities necessary to classify a new instance. From a machine learning perspective, we can define the Bayes' theorem as follows: given an unknown instance $A = (a_1, a_2, a_3..a_n)$, where $a_i(i = 1, \ldots n)$ are the values to the attributes of an instance, we wish to predict to which class does A belong to.

According to the Bayes' theorem, the probability of choosing a class given a sample is given by:

$$P(Class|A) = \frac{P(A|Class) \times P(Class)}{P(A)} \quad (1)$$

We can rewrite equation 1 in terms of the attributes of instance A:

$$P(Class|A) = \frac{P(Class|a_1, ..., a_n) \times P(Class)}{P(a_1, ..., a_n)} \quad (2)$$

The class definition for instance A involves the calculation of the probability for all possible classes given a particular attribute. The defined class is the one with the highest probability. From a statistical perspective, this is equivalent to maximising P(Class | a_1, Class | a_2, .., Class | a_n). Moreover, the denominator in equation 2 is constant, which leads to the following simplification:

$$argmax P(Class) \times \prod_i P(a_1, ...a_n|Class) \quad (3)$$

To perform the classification, we used data from a real experiment called "Reality Mining" [19].

2) Training data for the Naive Bayes classifier: In order to apply a machine learning technique, it is necessary to have a database for training and testing, including all attributes considered significant for the problem itself [20]. So, the database must include values for the attributes used as evidence for classification purposes. For example, if we want to classify the friendship strength among two nodes in a network. From a machine learning perspective, the class is the concept you want to learn, and the attributes are the evidences that can help at predicting the class. Consider that the evidences used in this problem and its possible values are: number of contacts (*weak*, *average*, *higher*), contacts duration (*small*, *average*, *large*) and contact frequency (*weak*, *higher*). Also, consider that the possible classes of the problem are: **weak** friendship (i.e., they are not friends) and **strong** (i.e., they are friends). The database is arranged in several tuples containing the values relative to each evidence used, and the class according to each evidence. For this database, each tuple will have a value to number of contacts, contacts duration and contact frequency, and a value for the class.

In this work, the reference database used for the friendship strength classification was taken from a MIT researchers experiment [19]. That project got the data from a test with mobile devices equipped with a Bluetooth network interface.The experiment collected data from 97 mobile phones over the course of six months. The scientific community has access to diverse information such as connectivity, proximity, location and activities of these users.

The main motivation for using this trace is the call history and text message among the users who participated in the experiments. This information is used as the basis for classifying the strength of the friendship among each pair of nodes in the network. Also, it is used to keep track of the number of meetings and contact duration among nodes. Thus, we model the friendship strength based on the following indicators:

- Frequency of meetings (FM) – among other factors, the friendship of two DTN nodes can be identified by a high frequency of meetings [2]. In this work, the FM represents how many times two nodes were within the radio range of each other. However, the higher the FM, the bigger will not be the friendship strength among the nodes.
- Contact duration (CD) – represents the average time that two nodes remain within each other's coverage area. However, although friends have usually long-lasting encounters, the same occurs among people that work together and are not necessarily friends.
- Amount of calls (AC) – represents the number of calls exchanged by a pair of nodes.
- Amount of text messages (ATM) – represents how many text messages a pair of nodes exchanged.

We model the levels of these indicators as follows:

- FM – weak, average, high.
- CD – weak, average, high.
- AC – weak, high.
- ATM – weak, high.

We model each characteristic used in this work based on values from Reality trace. For example, in Reality trace the FM is represented by numerics values (0 to 50 encounter times). Thus, we create three levels of FM and convert these numerical values into FM levels. For example, 0 to 4 frequency encounter can be converted into level weak, 5 to 15 into level average, and more than 15 into high.

Additionally, all interested researchers can access these traces to retrieve information such as the social relationship among people participating in the experiment. In our model we focused on answering the following question: "Is this person a part of your close circle of friends?". The answers can be "yes" or "no".

The class to be analyzed from the trace data is the friendship strength, which can be strong or weak. Thus, if node A answers "yes" to this question about node B, we assume in our training data that the friendship among A and B is *strong*.

If A answers "no" to the question, the friendship is defined as *weak*. Thus, the training data for the Naive Bayes algorithm consists of tuples with values relative to the aforementioned attributes, and to the friendship strength among the nodes. Table I presents an example of some tuples from the training data used in this work.

Algorithm 1 Step 1: Classification of friendship strength

```
1: procedure NAIVE BAYES(nodeA, nodeB)
2:     friendStrength ← 0
3:     probIsStrong ← calcProb(historyDataFromNodes)
4:     probIsWeak ← calcProb(historyDataFromNodes)
5:     if probIsStrong >probIsWeak then
6:         friendStrength ←strong
7:     else
8:         friendStrength ←weak
9:     end if
10: end procedure
```

TABLE I
A SAMPLE OF THE TRAINING DATA USED IN THIS WORK

FM	CD	AC	ATM	Strength
weak	no	no	high	weak
weak	no	no	weak	weak
average	no	no	high	weak
high	yes	yes	high	weak
high	yes	yes	weak	strong
high	yes	yes	weak	strong
average	yes	no	high	weak
weak	no	yes	high	weak
high	yes	yes	weak	strong
average	no	no	weak	weak
average	no	no	high	strong
high	yes	yes	high	weak
weak	yes	yes	high	strong
average	yes	no	weak	weak
weak	yes	yes	high	weak

3) Example of Naive Bayes classification: To clarify our approach, we present an example. Consider that at a given time, nodes A and B are in the coverage area of each other. Also, consider that node B carries a message to node C. Consider from the contact history of A and C the following instance *FM = high, CD = high, AC = no, ATM = yes*. FSF has to decide whether to forward the message to node A. To this end, FSF asks the Naive Bayes algorithm for the friendship strength among A and C. To answer this question, Naive Bayes follows the steps below:

Step 1 – Determine the probability of occurrence of each class:

$$P(Class) = \frac{P}{Q} \tag{4}$$

where P is the number of cases of class X, and Q is the total number of cases.

P(Strength = weak) = 10/15 = 0.66

P(Strength = strong) = 05/15 = 0.34

Step 2 – Determine the probability of the attributes for the instance in question about every possible class:

$$P(Attribute(Z)|Class) = \frac{R}{S} \tag{5}$$

where R is the total number of cases of class Y concerning characteristic A_i, S is the number of cases of class Y, and Z is the value for characteristic A_i.

P(FM = high AND strength = strong) = 2/5 = 0.4

P(FM = high AND strength = weak) = 3/10 = 0.3

P(AC = no AND strength = strong) = 2/5 = 0.4

P(AC = no AND strength = weak) = 4/10 = 0.4

P(ATM = yes AND strength = strong) = 4/5 = 0.8

P(ATM = yes AND strength = weak) = 4/10 = 0.4

P(CD = high AND strength = strong) = 4/5 = 0.8

P(CD = high AND strength = weak) = 5/10 = 0.5

Step 3 – Determine the probability of each class based on the probability of the instance:

$$P(Class|a_1, a_2, ..., a_n) = \prod P(a_i|Class) \times P(class) \tag{6}$$

P(A|class = strong) = (0.4 * 0.4 * 0.8 * 0.8 * 0.34) = 0.034

P(A|class = weak) = (0.3 * 0.4 * 0.4 * 0.5 * 0.66) = 0.015

In this example, and according to the calculated probabilities, the friendship strength among nodes A and B would be classified as *strong*.

B. Classification of the individual selfishness

Now that FSF has evaluated the friendship strength, FSF evaluates the individual selfishness from the message relay point view. According to [9], this individual selfishness can be defined as: "the unwillingness of a single node to relay the messages of all other nodes in order to conserve its limited resources". In the DTN literature, most authors consider that the stronger the social relationship with a node is, the more the willingness to help it. Despite we clearly agree with the logic behind this assumption, we nevertheless believe that, in some situations, the message relay cannot handle more messages because its resources are at critical levels. In such cases, the message relay will *priorities* instead, to save its resources.

In this work we propose a model that takes into account two fundamental resources: battery level and memory availability. When a contact opportunity arises, if the friendship strength among the nodes is considered strong, FSF checks the individual selfishness of the message relay to decide whether to forward the message. Hence, any message will be forwarded if, and only if, the available resources at the message relay device have not reached critical levels.

From the example in section A.3, the friendship strength among nodes A and C was classified as strong. However, lets now assume that the device carried by node A has only 50% of battery energy level and 70% of used memory space. In such a case, it is reasonable to assume that A, the candidate node to relay the message, does not accept the message to node C because it has to save its mobile resources.

To simulate this situation we consider that, when a contact opportunity arises, if the message relaying device has an energy level less than α, or, if the used memory space is more

TABLE II
SOME PARAMETERS ABOUT TRACES UTILIZED IN THE SIMULATION

Trace	*Reality*
Device	*Phone*
Network Interface	*Bluetooth*
Nodes	97
Trace Duration	246 days
Granularity	300 s
Number of contacts	54667

than β, the node will not accept the message, although the friendship among the corresponding nodes may be strong. For the sake of experimentation, in our simulations we consider $\alpha = 30\%$ e $\beta = 70\%$.

C. Forwarding Strategy

After explaining the two tasks performed by FSF, we can define the forwarding strategy. If a node B having a message destined to node C meets with node A, node B will forward the message destined to node C to node A if, and only if, the friendship strength among nodes A and C is strong, and if the device resources of node A have a battery level above α, and its memory occupation is less than β. If node B forwards the message to node A, the message will not be deleted from the buffer of node B to increase the message delivery probability. It is worth mentioning that, if the friendship strength among nodes A and C is weak and/or the device resources of node A are at critical levels, the message will not be forwarded to node A.

IV. EVALUATION METHODOLOGY

To evaluate our FSF algorithm, we performed trace-driven simulations using the "ONE Simulator" [21]. To simulate node mobility we used a real mobility trace called Reality, derived from the Reality Mining project [19].

We implemented a simulation model which includes FSF and, for the sake of performance comparison, we also implemented two well-known algorithms found in the literature: BUBBLE-RAP [8] and Friendship routing (FS) [7]. The BUBBLE-RAP algorithm forwards messages by taking into account node popularity and its community information. We selected this algorithm for being well known in the DTN literature, and because it also adopts a strategy that accounts for social characteristics. In turn, the FS algorithm forwards messages by taking into consideration the value of a metric called Social Pressure Metric (SPM), based on characteristics like frequency of contacts, longevity, and regularity. FS is considered the first routing mechanism based on the friendship level among DTN nodes.

Due the constraints in terms of DTN node capacity, we had to select a buffer management policy to deal with overflow conditions. To avoid favouring the routing decisions of the algorithms under comparison, we selected the Drop Oldest buffer management strategy, which discards the oldest message in the case of buffer overflow.

TABLE III
SOME PARAMETERS DEPLOYED IN THE SIMULATION

Transmission speed	250 kbps
Message generation rate	2 msg/sec
Message size	0,5 - 1 MB
TTL	5 h

The DTN performance can be measured in terms of average delivery ratio, and average delivery delay [22]. Moreover, we applied two other metrics widely used in other works found in the literature, i.e. [7], [5] and [6]:

- Average delivery ratio – the ratio of messages received by the destination nodes to those generated by the source nodes.
- Average delivery delay – the average time interval among the sending and receiving events for messages traveling along the network.
- Average cost – the average number of forwarding events per message delivered to the destination.
- Efficiency – the ratio among delivery ratio and average cost.

Table II and Table III list other simulation parameters used in our simulation experiments. Messages were generated every two seconds, and a pair of source/destination nodes were chosen randomly.

To simulate the power consumption of the nodes, we adopted the energy model proposed in [23]. The power consumption of a node is classified into five states:

- Off – no power consumption as the network node interface is turned off.
- Inactive – reduced power consumption as the network node interface is idle.
- Scan – the node consumes power while the network node interface detects neighbors.
- Transmission – the node consumes power while sending a message.
- Reception – the node consumes power because it is receiving a message.

In our simulations, we assume that the nodes initially have a maximum level of energy. We consider that the energy level of the nodes is measured in units, being 500 units the highest value. We assume that the user recharges his/her device every 24 hours. Energy consumption depends on the device's state and the number of operations using the network interface. For example: if the node is at transmission, reception or scan states, we assume a reduction of 25 energy units for every sent/received message and/or for every scan performed. If the node is at off or inactive states, we assume there is no energy cost.

V. RESULTS

In this section we present and discuss the results, as well some important questions from our proposal as the impact of individual selfishness on routing and an analysis of the classification from the machine learning algorithm.

TABLE IV
COMPARISON OF DELIVERED MESSAGES PER CLASS.

	Created	Delivered
Friends	3853	2017
Not friends	7159	1578

A. Delivery Ratio

Figure 1 presents the results for the delivery ratio obtained in the Reality scenario when increasing the node buffer size from 10 to 80 MB (similarly to other works in the literature [22], [24], [25]). We find that FSF outperforms the two other algorithms by delivering more messages. It is caused by the consideration of friendship in routing decisions. FSF consider that people with friendship bonds have more willingness to help each other, and FSF attempts to concentrate on these users the responsibility for carrying a message to increase the delivery probability and save node resources. However, for small buffer sizes, the Friendship routing strategy provided an almost similar performance, but performance differences become more evident as the buffer size increases. Notice that, the bigger the buffer size, the more messages can be stored, and so there are less chances of buffer overflow. Thus, fewer messages will be dropped, and can be forwarded in future contact opportunities.

In general, a smaller buffer size potentially favors buffer overflow, causing the removal of old messages to store new ones. This drop policy has a negative impact on the FSF delivery ratio because messages addressed to nodes with strong friendship can still be dropped. A possible solution to this issue is to increase the number of classes to differentiate among different degrees of friendship. For instance, we could assume three classes: weak, average and strong. This way, FSF could firstly discard messages addressed to nodes with weak friendship strength, followed by destinations with average friendship strengths, thus allowing that messages addressed to nodes with a high friendship strength to experience a higher delivery probability.

Table IV presents the ratio of created and delivered messages by the FSF algorithm, taking into account the friendship strength among the source and destination nodes. About 54% of the messages addressed to source/destination nodes with strong friendship were delivered, while this ratio was decreased to 20% for source/destination nodes with a weak friendship. This information reinforces the importance of using friendship strength as an appropriate criterion in DTN message routing.

B. Average Delay

It is important to minimize the average delivery delay in DTN. Figure 2 presents the results for the average delivery delay in the Reality scenario. The FSF algorithm presented a reasonable behaviour concerning the average delay to deliver messages, although the time required for that task was somewhat higher compared to the Friendship Routing protocol. We can justify it by a higher intercontact time among the nodes that have a strong friendship. The bigger the intercontact among these nodes, the higher the average delay, because the relay will need of longer time to deliver the message to the destination. Thus, FSF can be negatively impacted if the intercontact time among nodes with a strong friendship is high. We are investigating manners to improve FSF performance in such situations by using more classes of friendship. Both algorithms based on friendship achieved smaller average delivery delay when compared to BUBBLE-RAP, that does not use this criterion.

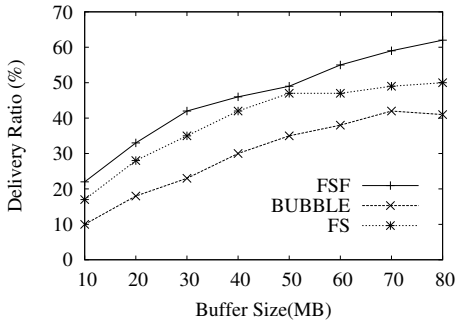

Fig. 1. Delivery ratio in the Reality scenario.

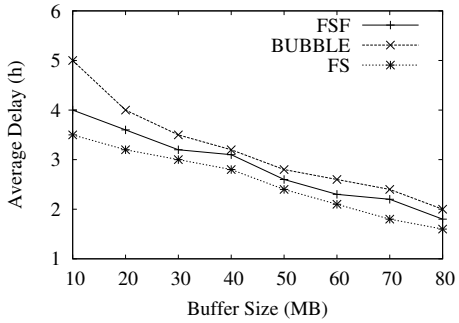

Fig. 2. Average delay in Reality scenario

C. Average Cost

The average cost is the average number of forwards done per message during the simulation. The lower the average cost, the better the performance, as all forwarding events consume resources from the nodes involved in the relaying process. Nevertheless, by reducing the number of copies of a message, we reduce the delivery probability. FSF attempts to find a balance among the need for forwarding to improve the delivery ratio, and minimising transmissions to save resources. Figure 3 presents the average cost results obtained in the Reality scenario. FSF achieves the best result, lowering the average number of forwarding per delivered message compared to the other algorithms evaluated. It is caused by the less number of message' forwards to deliver a message to the destination. The bigger the node' willingness to help a node, the greater the delivery probability. In our algorithm, the message will forward to relay if, and only if, the friendship among the relay

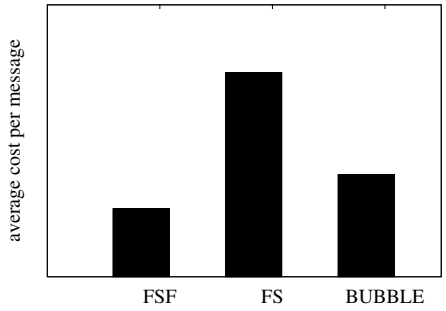

Fig. 3. Average cost in Reality scenario.

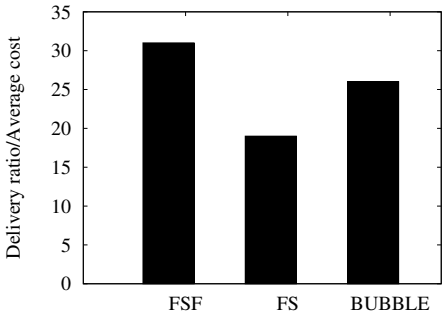

Fig. 4. Efficiency in the Reality scenario.

and the message' destination is strong. It cause the decreased of number of message' forwarding.

D. *Efficiency*

Efficiency is an indicator of network resource usage. The greater the efficiency, the better the utilization of the network resources. Even with a high delivery ratio, a routing algorithm with very low average cost in terms of forwarded messages cannot be considered an efficient algorithm if it consumes too many network resources.

Figure 4 presents the results for the efficiency metric in the Reality scenario. FSF achieved the best results with an average cost of 2 forwarded messages.

The FSF's effectiveness becomes evident due to the smaller number of forwardings required for delivering a message to each recipient. This occurs because FSF attempts to concentrate on users that have a strong friendship with the message destination, and so it is reasonable to assume that the bigger the friendship strength, the greater the message deliver probability. FSF selects those nodes with a strong friendship with the message destination to take the responsibility for carrying and delivering its messages. In addition, to increase the message delivery probability while saving the node resources, less message forwarding events will be required to deliver a message.

E. Additional Considerations and Discussion

To analyze the impact of individual selfishness on network performance, we monitored all the messages that have been created. Then, we compared the number of messages refused by the candidate nodes, and the number of messages not delivered to the destination node. Table V shows the percentage of messages not received from the messages not delivered. As we can see from these results, FSF did not deliver up to 65% of the messages due resource constraints. Although we can not assert that the messages not received by some node were not delivered due to this particular reason, it is reasonable to assume that there may be a correlation among these factors.

TABLE V
COMPARISON AMONG MESSAGES NOT DELIVERED AND MESSAGES NOT RECEIVED DUE TO RESOURCE CONSTRAINTS.

Buffer Size	Not delivered	Not received	%
10MB	1817	713	39
20MB	1622	699	43
30MB	1489	741	49
40MB	1211	790	65
50MB	1000	552	50
60MB	703	338	48
70MB	579	374	64
80MB	478	287	60

An important issue affecting the performance of the FSF algorithm is the correct classification of the friendship among nodes in the network. In this work, we introduced an approach to classifying the friendship strength using a machine learning algorithm. This algorithm has two important tasks: first, to learn what friendship is based on data collected from the real world. The second task is to classify new relationships among two nodes in the network. From the FSF results, it is reasonable to assume that Naive Bayes offers an excellent performance at classifying the friendship among nodes. So, the better the classification from the Naive Bayes algorithm, the better the FSF performance.

The deployment of a machine learning algorithm proved to be an interesting solution for the task of classifying friendship strength among two nodes. However, some issues must be taken into account. For example, the availability of information about what is friendship in the real world can be a strong requirement since, in some scenarios, this information is not available. To solve this problem, one can monitor the scenario through an application responsible for collecting information about the friendship among nodes. Another alternative is to use a machine learning algorithm that does not need a training database to classify new instances.

Finally, we consider that the use of a machine learning approach can be more flexible than other approaches for classifying the friendship among nodes. For example, if we need to change the friendship model features, using machine learning really accelerates this process. Another advantage is that a machine learning approach is able to combine several characteristics to classify the friendship strength, and so it can easily use data from real world situations.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we proposed the Friendship and Selfishness routing (FSF) algorithm for DTN environments. FSF considers

two social characteristics: the friendship and the individual selfishness of the nodes candidate to relaying messages. FSF carries out two main tasks: firstly, FSF uses a machine learning technique to classifying the friendship strength among nodes; secondly, FSF determines the individual selfishness by taking into account both the energy level and the secondary memory availability of each node. The goal is address those cases when the node rejects the message despite it has a strong friendship with the destination node.

To validate our proposal, we conducted a set of experiments to determine the message delivery effectiveness in a DTN scenario based on trace-driven simulations [19]. Compared to two other routing algorithms in the same category, FSF offered better results regarding a set of standard metrics, namely the delivery ratio, the average cost, and efficiency, while providing similar delivery delays.

As future work, we intend to carry out a comparative study of adopting other machine learning algorithms to classify the friendship strength among pairs of nodes. Moreover, we want to analyze the impact of selfishness in the routing process by using different thresholds for device resource restrictions.

VII. ACKNOWLEDGMENTS

This work was partially supported by the "Programa Estatal de Investigacio'n, Desarrollo e Innovacio'n Orientada a Retos de la Sociedad, Proyecto I+D+I TEC2014-52690-R".

REFERENCES

[1] C. T. de Oliveira, M. D. Moreira, M. G. Rubinstein, L. H. M. Costa, and O. C. M. Duarte, "Redes tolerantes a atrasos e desconexões," *SBRC Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, 2007.

[2] Y. Zhu, B. Xu, X. Shi, and Y. Wang, "A survey of social-based routing in delay tolerant networks: positive and negative social effects," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 1, pp. 387–401, 2013.

[3] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[4] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, "Maxprop: Routing for vehicle-based disruption-tolerant networks," pp. 1–11, april 2006.

[5] Q. Li, S. Zhu, and G. Cao, "Routing in socially selfish delay tolerant networks," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[6] Y. Qin, L. Li, X. Zhang, and X. Zhong, "Nfcu: A new friendship-based routing with buffer management in opportunistic networks," *arXiv preprint arXiv:1501.07754*, 2015.

[7] E. Bulut and B. K. Szymanski, "Friendship based routing in delay tolerant mobile social networks," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*. IEEE, 2010, pp. 1–5.

[8] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: Social-based forwarding in delay-tolerant networks," *Mobile Computing, IEEE Transactions on*, vol. 10, no. 11, pp. 1576–1589, 2011.

[9] J. Miao, O. Hasan, S. B. Mokhtar, L. Brunie, and K. Yim, "An investigation on the unwillingness of nodes to participate in mobile delay tolerant network routing," *International Journal of Information Management*, vol. 33, no. 2, pp. 252–262, 2013.

[10] Y. Li, P. Hui, D. Jin, L. Su, and L. Zeng, "Evaluating the impact of social selfishness on the epidemic routing in delay tolerant networks," *Communications Letters, IEEE*, vol. 14, no. 11, pp. 1026–1028, 2010.

[11] Y. Li, G. Su, D. O. Wu, D. Jin, L. Su, and L. Zeng, "The impact of node selfishness on multicasting in delay tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. 60, no. 5, pp. 2224–2238, 2011.

[12] Q. Li, W. Gao, S. Zhu, and G. Cao, "A routing protocol for socially selfish delay tolerant networks," *Ad Hoc Networks*, vol. 10, no. 8, pp. 1619–1632, 2012.

[13] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22. IBM New York, 2001, pp. 41–46.

[14] J. Gama and O. teorema de Bayes, "Aprendizagem bayesiana introdução," *Universidade do Porto*, pp. 1–22, 2012.

[15] D. D. Lewis, "Naive (bayes) at forty: The independence assumption in information retrieval," in *Machine learning: ECML-98*. Springer, 1998, pp. 4–15.

[16] M. Panda and M. R. Patra, "Network intrusion detection using naive bayes," *International journal of computer science and network security*, vol. 7, no. 12, pp. 258–263, 2007.

[17] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine learning*, vol. 29, no. 2-3, pp. 103–130, 1997.

[18] H. A. Simon, "Why should machines learn?" in *Machine learning*. Springer, 1983, pp. 25–37.

[19] A. Pentland, N. Eagle, and D. Lazer, "Inferring social network structure using mobile phone data," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 106, no. 36, pp. 15 274–15 278, 2009.

[20] M. J. Amorim, D. Barone, and A. U. Mansur, "Técnicas de aprendizado de máquina aplicadas na previsao de evasao acadêmica," *XIX Simpósio Brasileiro de Informática na Educação*, pp. 12–14, 2008.

[21] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. New York, NY, USA: ICST, 2009.

[22] G. Fathima and R. Wahidabanu, "Buffer management for preferential delivery in opportunistic delay tolerant networks," *International Journal of Wireless & Mobile Networks (IJWMN)*, vol. 3, no. 5, pp. 15–28, 2008.

[23] D. R. Silva, A. Costa, and J. Macedo, "Energy impact analysis on dtn routing protocols," 2012.

[24] A. Krifa, C. Baraka, and T. Spyropoulos, "Optimal buffer management policies for delay tolerant networks," in *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON'08. 5th Annual IEEE Communications Society Conference on*. IEEE, 2008, pp. 260–268.

[25] Y. Li, M. Qian, D. Jin, L. Su, and L. Zeng, "Adaptive optimal buffer management policies for realistic dtn," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*. IEEE, 2009, pp. 1–5.